



# **Lightweight Programming for VR: Towards a Virtual Laboratory**

***Luc Renambot***

***Henri Bal***

***Hans Spoelder***

***Desmond Germans***

***vrije Universiteit - Amsterdam***

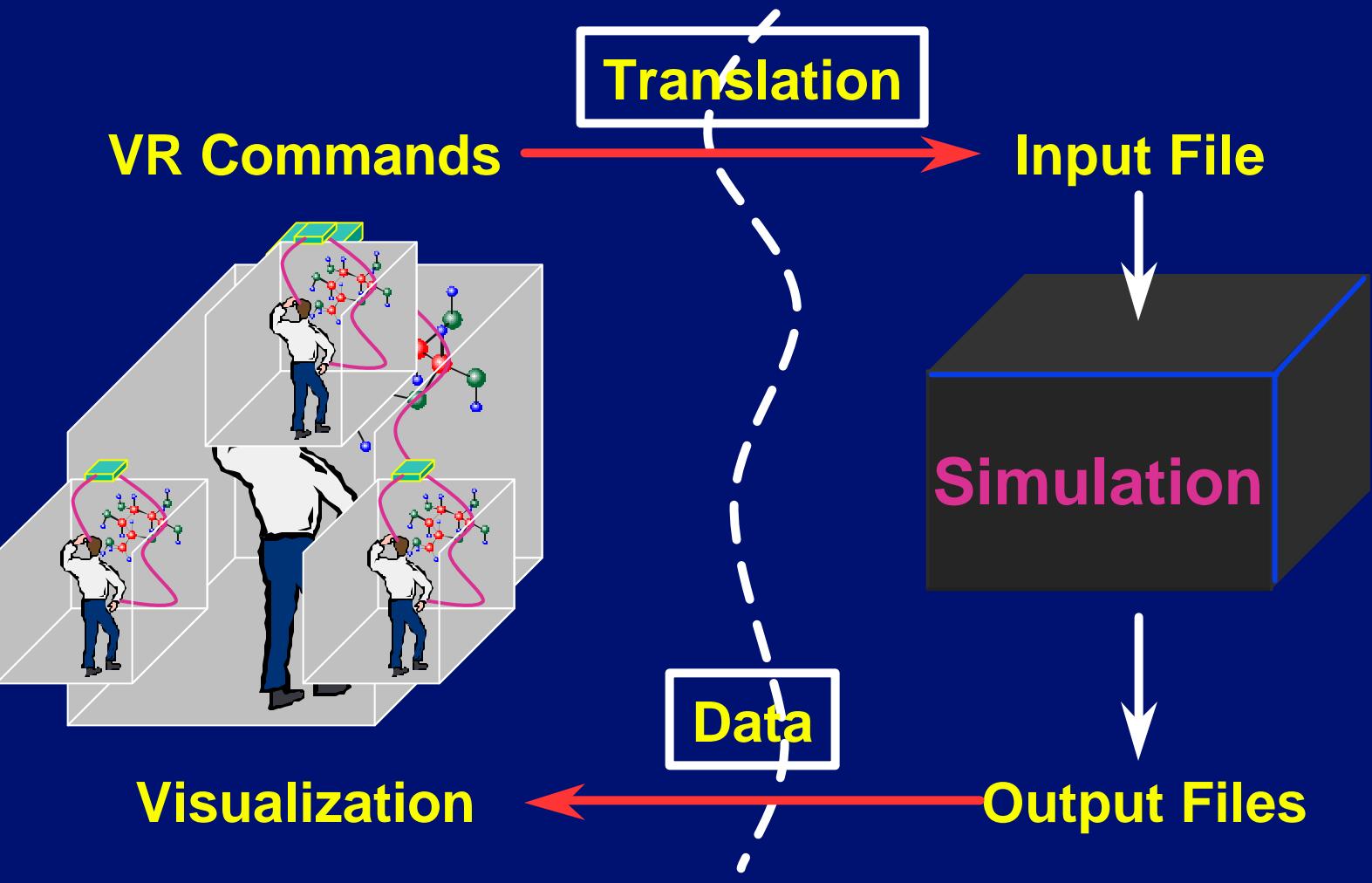


# Objectives

- ***Ease computational steering***
  - No source modification
  - From various VR systems
  - For the non-expert
- ***Collaboration through simulation or dataset***
- ***Interaction and Measurement in data space***

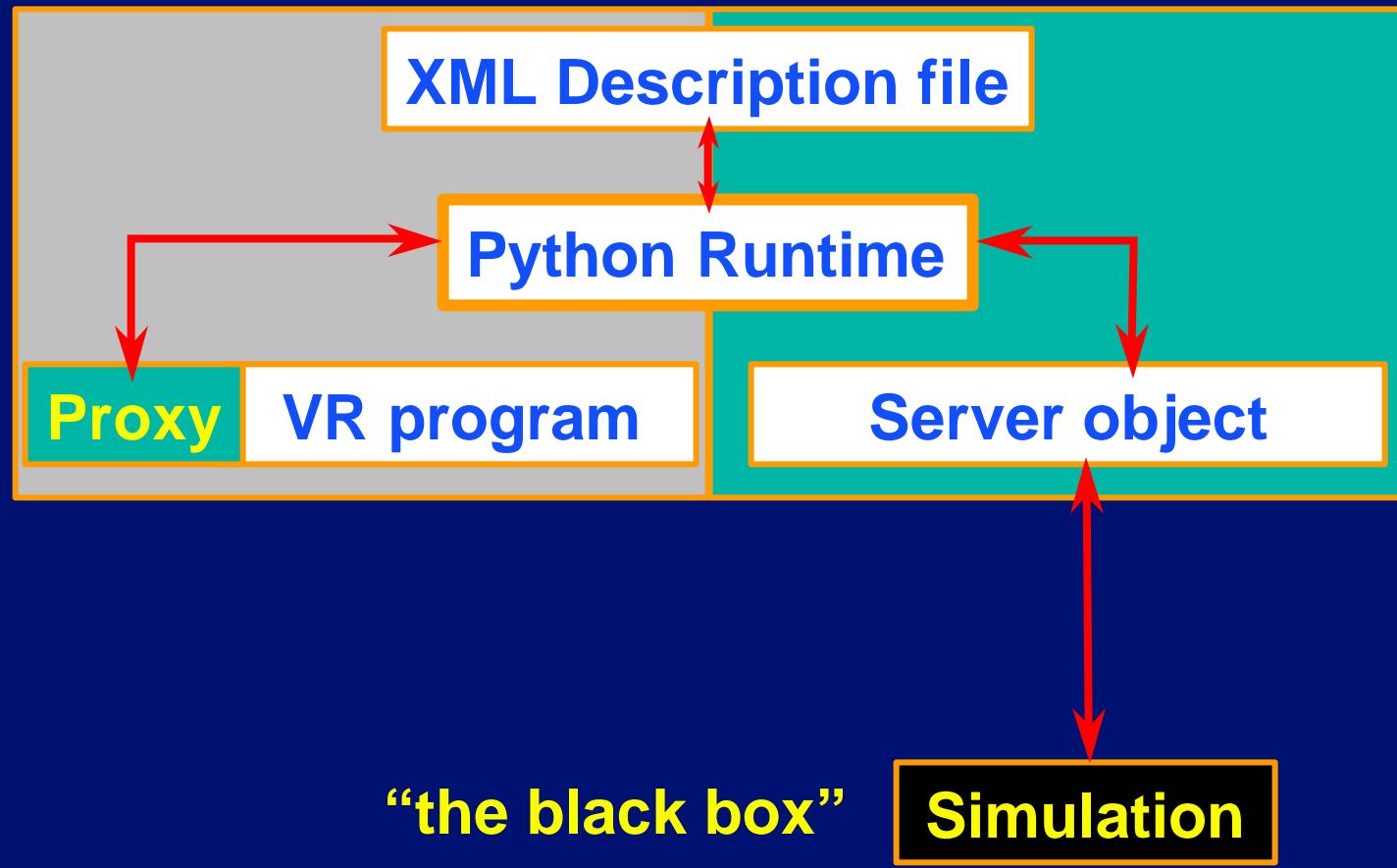


# Interactive Visualization





# CAVE Study



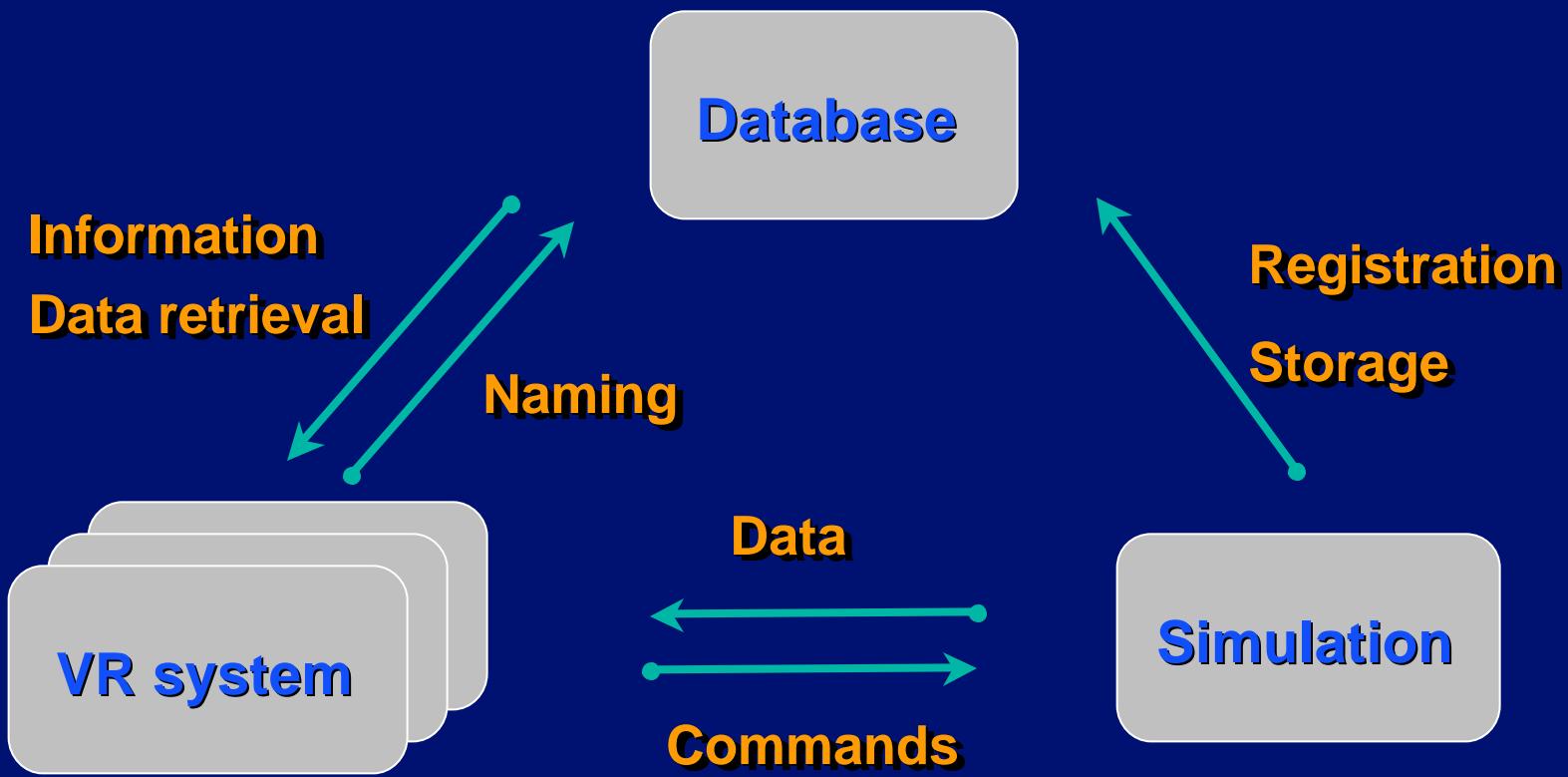


# Infrastructure

- ***Computational steering***
  - Simulation description in XML
  - Control as Python script and CAVERNSSoft G2
- ***VR***
  - Aura : portable and efficient 3D layer
  - VIRPI : high-level VR toolkit (C++/Python binding)
- ***Persistence***
  - Database : Information storage (meta)
  - Data repository : access through URL

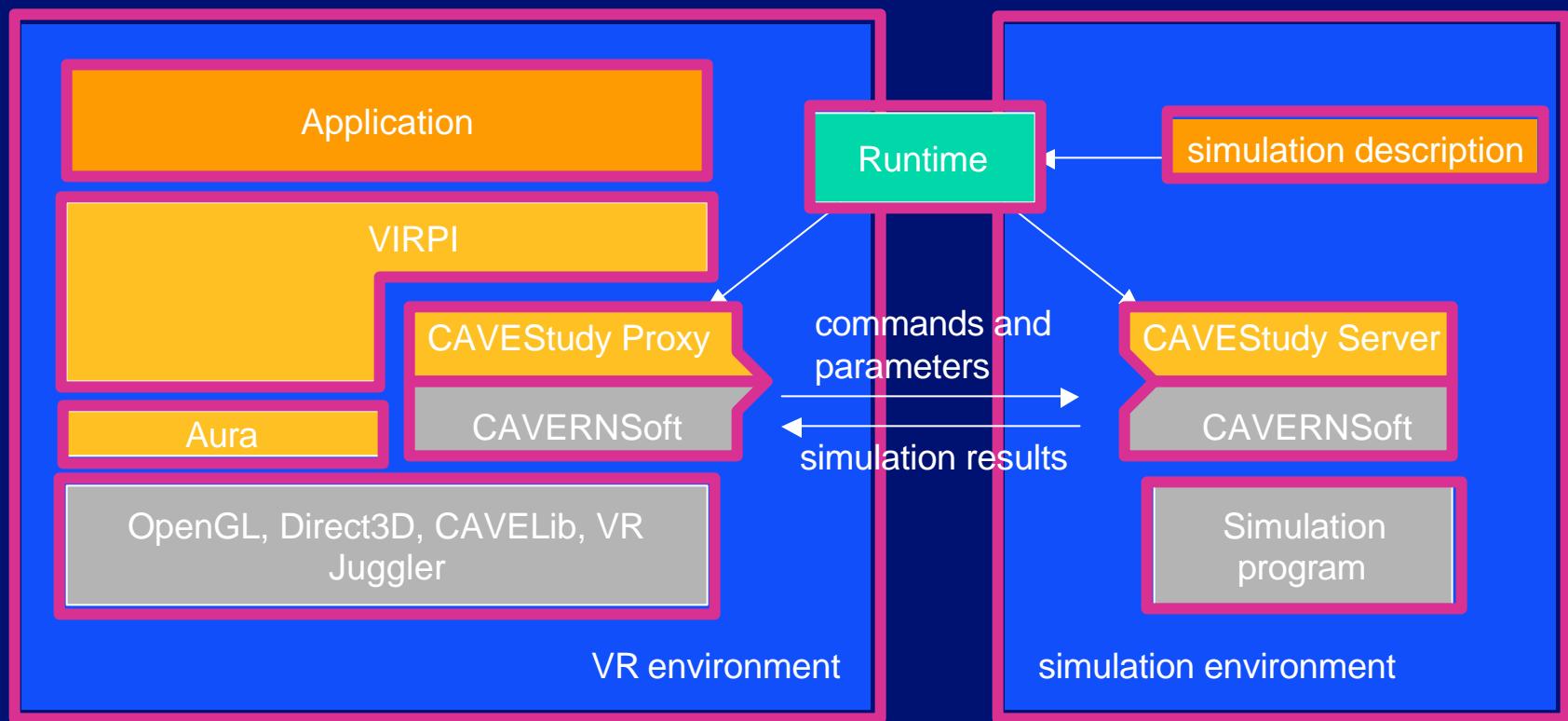


# Components



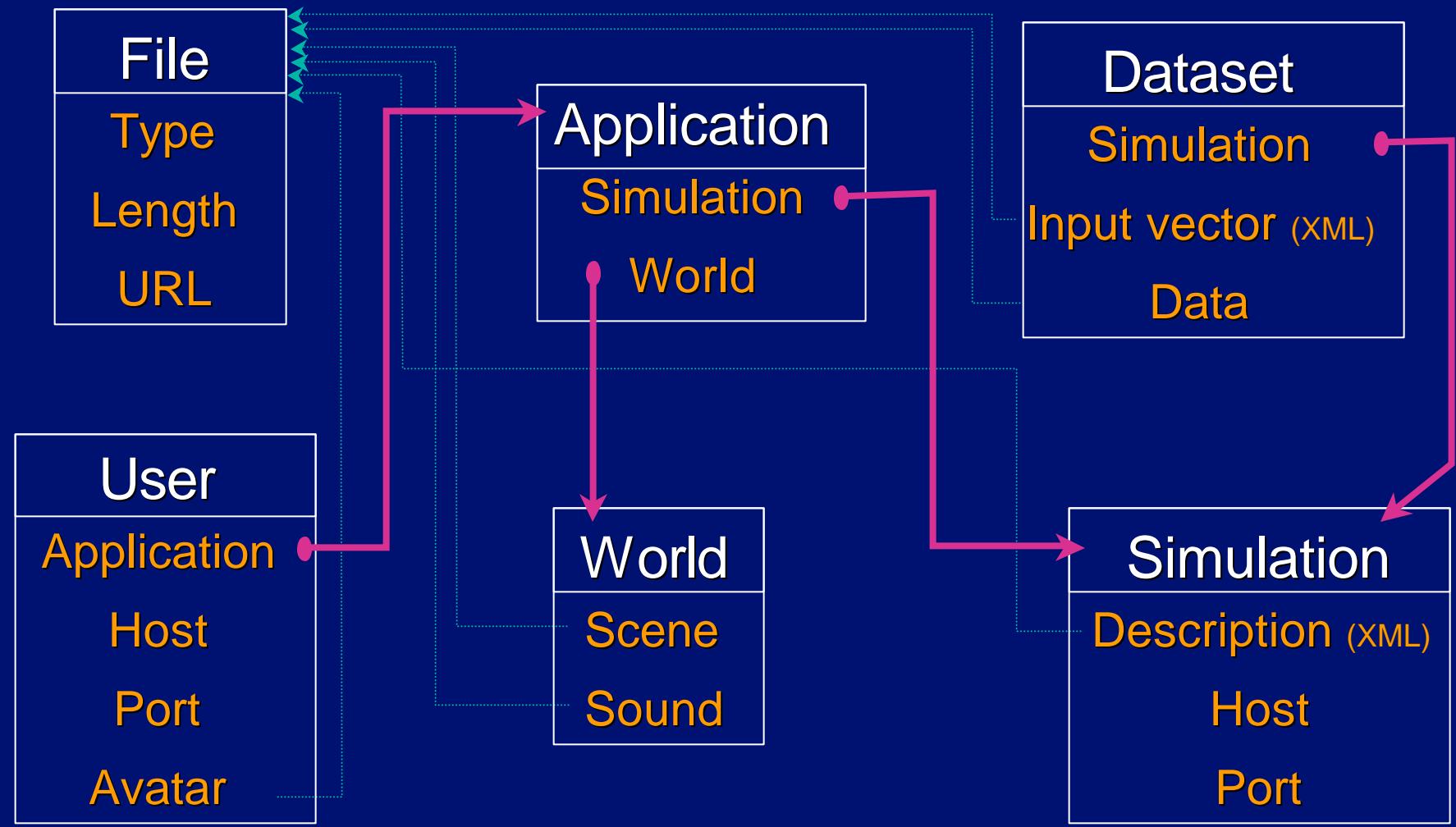


# Software Layers





# Database Schema





# XML Description

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!DOCTYPE CAVEStudy SYSTEM "cavestudy.dtd">

<CAVEStudy id="Sisyphus">

<struct id="vector4D">
    <field id="pos" type="float" array="3" />
    <field id="t" type="float" array="1" />
</struct>

<simulation id="filsim">
    <directory value="filsim" />
    <executable value="filsim" />
    <processes value="1" />
    <in type="commandline" />
    <out type="file" default="data.fil"/>
</simulation>
```

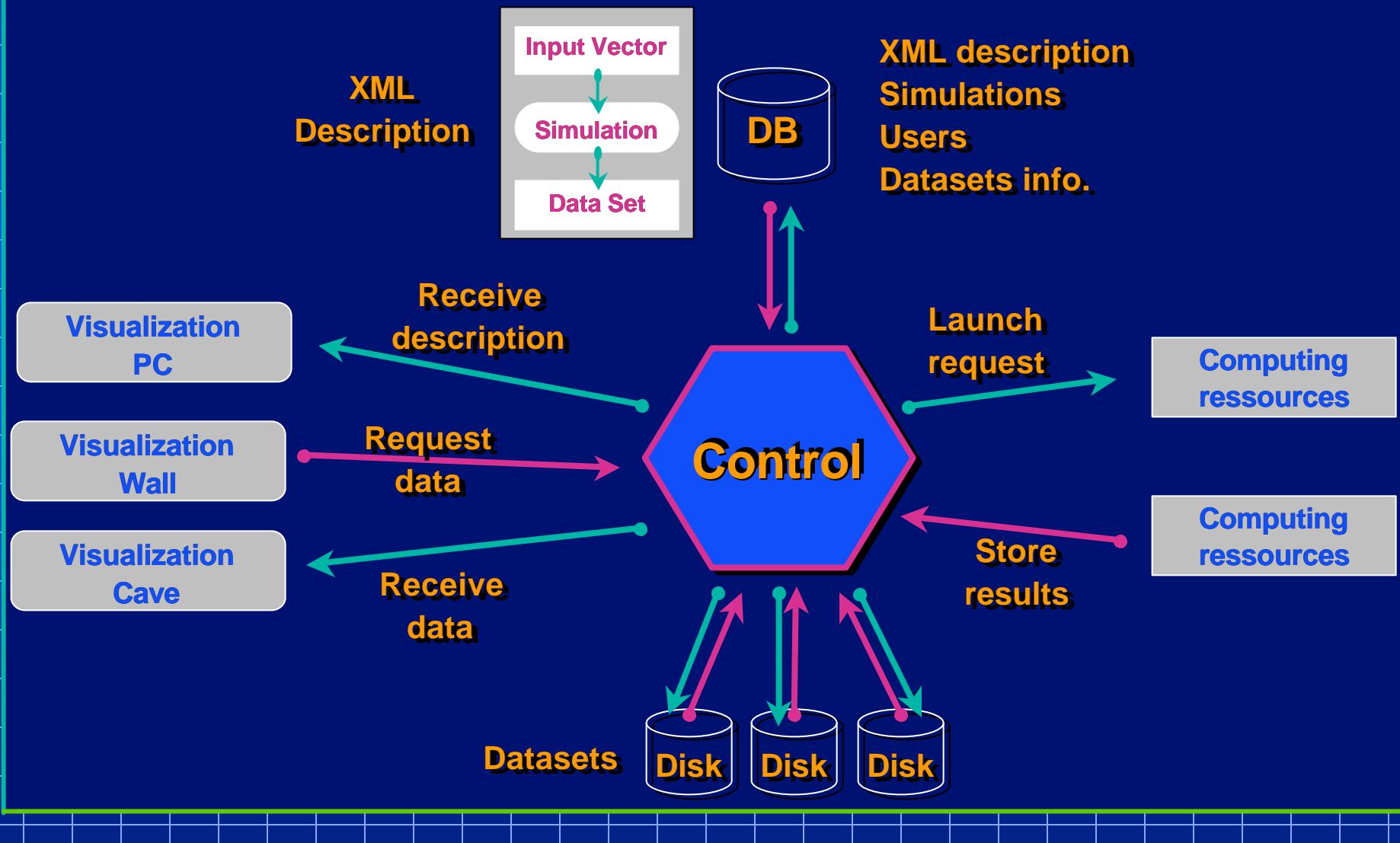
```
<input id="alpha" type="float"
       value="5.0" dim="0" />
<input id="pump_current" type="float"
       value="140254.6" dim="0" />
<input id="feedback_rate" type="float"
       value="0.00782" dim="0" />
<input id="starting_fp" type="int"
       value="0" dim="0" />

<output id="fixed_point" type="vector3D"
        dim="1" />
<output id="new_point" type="vector4D"
        dim="0" />

</CAVEStudy>
```

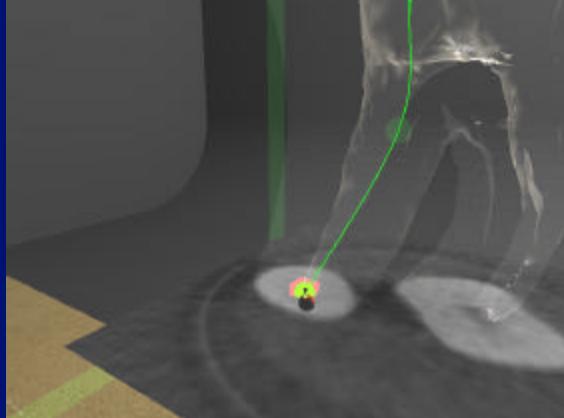


# XML as a Glue

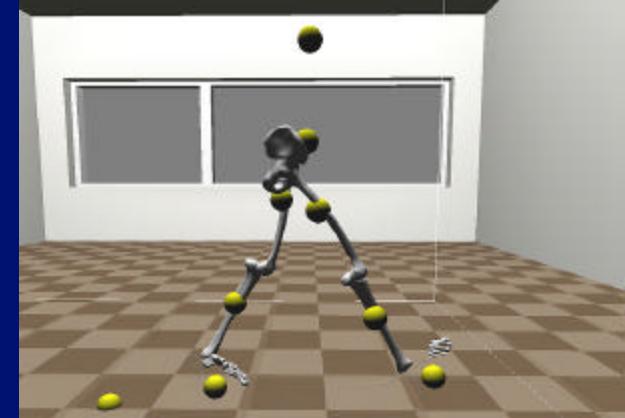




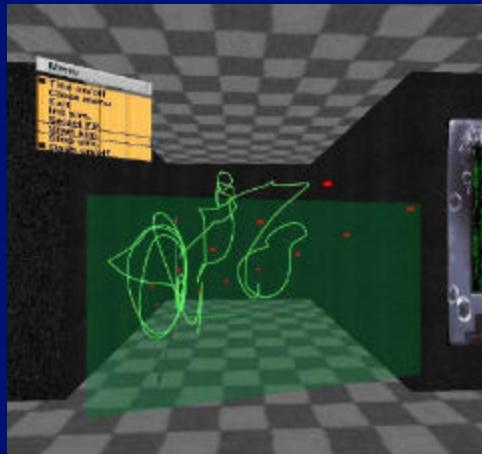
# Case Studies



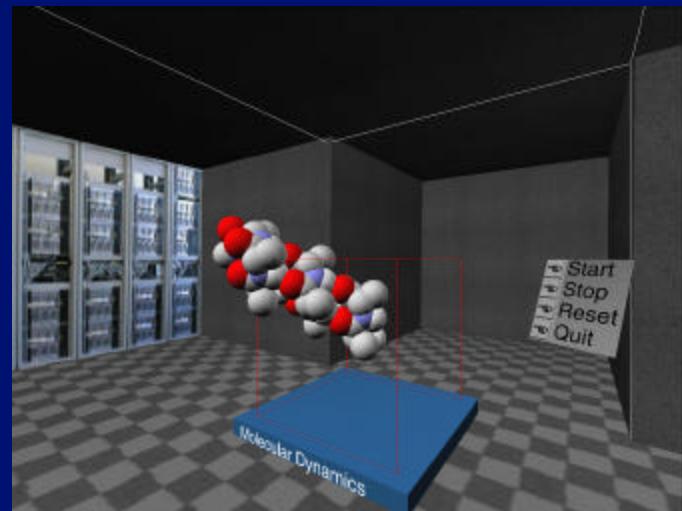
*Dentistry*



*Motion  
Science*



*Theoretical  
Physics*

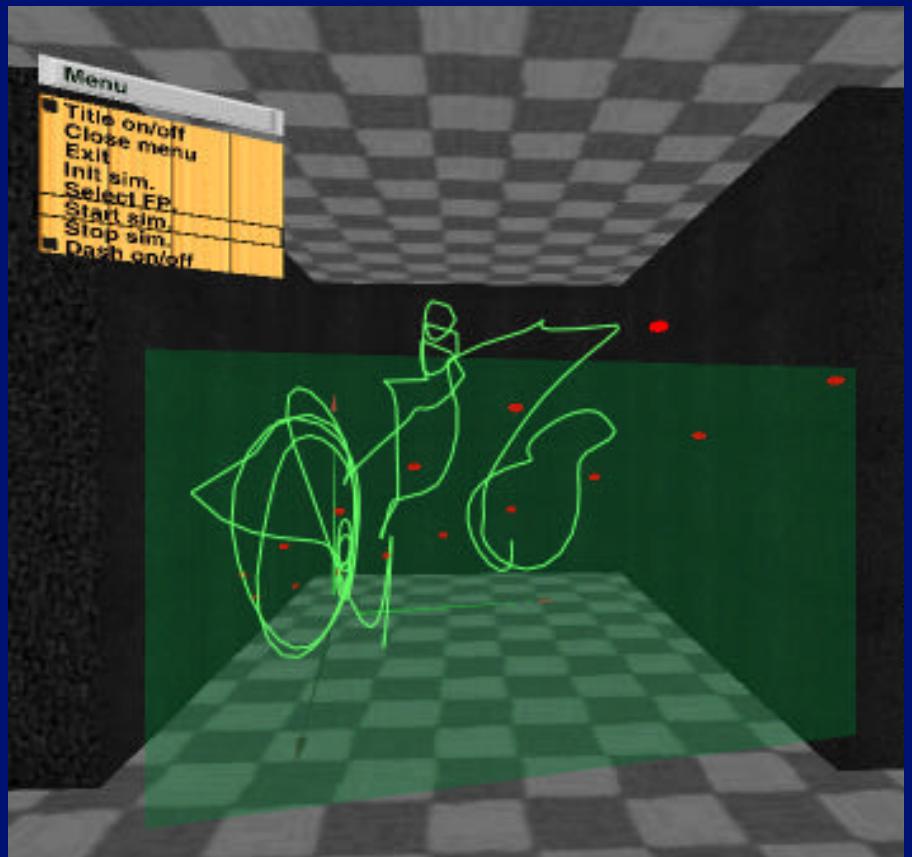


*Biochemistry*



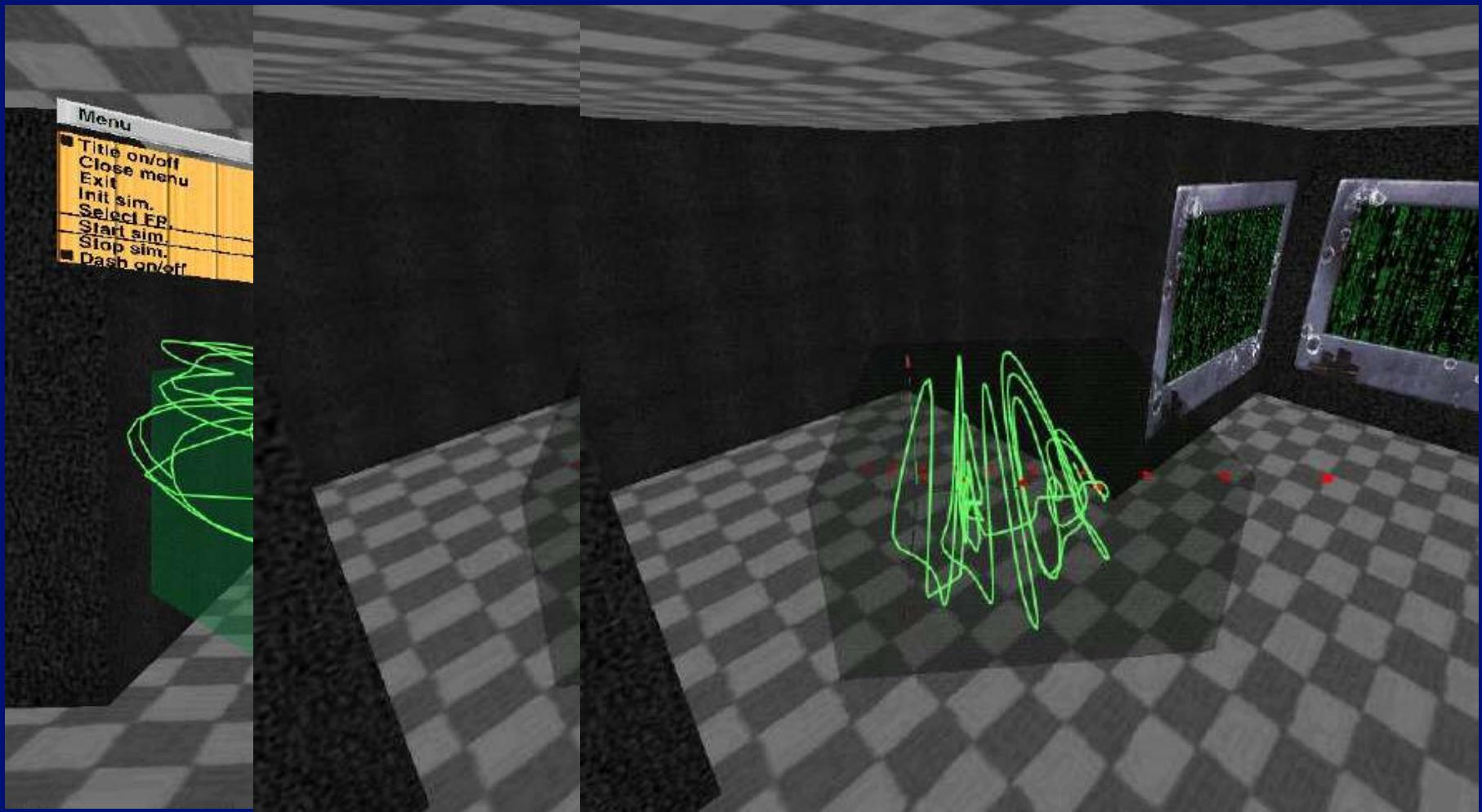
# Applications: Laser simulation

- ***Numerical simulation  
for a diode laser with  
optical feedback***
- ***Chaotic system***
- ***Interactive exploration  
of the phase space***





# Laser simulation





# Laser simulation

## ■ ***Study***

- Effect of 3 main parameters
- Initial-condition sensitivity

## ■ ***Benefit***

- Interactive modification of parameters
- Simulation program under development
- Better insight of the complex system
- XML : 30 lines, Server : 50 lines, VR : 300 lines



# Conclusion

- ***Steering environment in VR***

- XML description
- Python scripting
- Database storage

- ***Future***

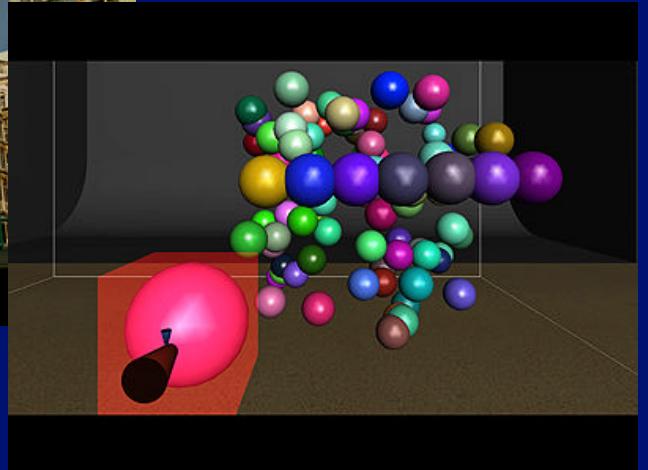
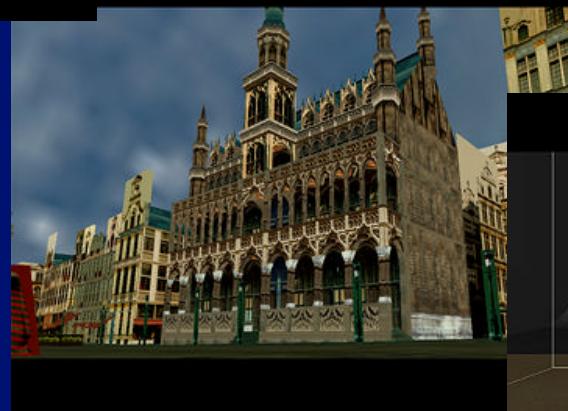
- Complete it: sync between user, smart download, ...
- Integration of AG technologies
- Port to a cluster for tiled display



# Information

***<http://www.cs.vu.nl/~renambot/vr>***  
***[renambot@cs.vu.nl](mailto:renambot@cs.vu.nl)***

# Gallery





# Scripts : connection

# **Connection to MySQL**

```
sql = cDatabase()
```

# **Get the xml file**

```
description,host,port =  
    sql.GetSimulation("Sisyphus")
```

# **Download the XML description**

```
sql.retrieveFile(description,  
    "simul.xml", "XML")
```

# **Build an XML parser**

```
parser= XmlParser(CavestudyXML())
```

# **Parse the downloaded XML file**

```
parser.Parse("simul.xml")
```

# **Generate the client program**

```
parser.Generate()
```

# **Load the generated module**

```
module = __import__("Sisyphus")
```

# **Create a client to the remote server**

```
client = module.Create(host,port)
```

# **Set an input parameter**

```
client.data["alpha"] = 4.5
```

```
client.send_alpha()
```

# **Start**

```
client.start()
```



# Measuring in VR

- ***Bring measuring paradigm to VR***
  - Human in the Loop
- ***Measure in visualization space***
  - No change to simulation program
  - Rapid and flexible experimental ‘prototyping’ to count, sum, average, derive, etc. quantities



# Aura

- ***Provide abstraction over VR system***

- scene graph, state management
- various standards:
  - *OpenGL, Direct3D, CAVELib, etc.*
- texture loading and management
- easy and direct C++ API
- Python binding
- Cluster version under dev.



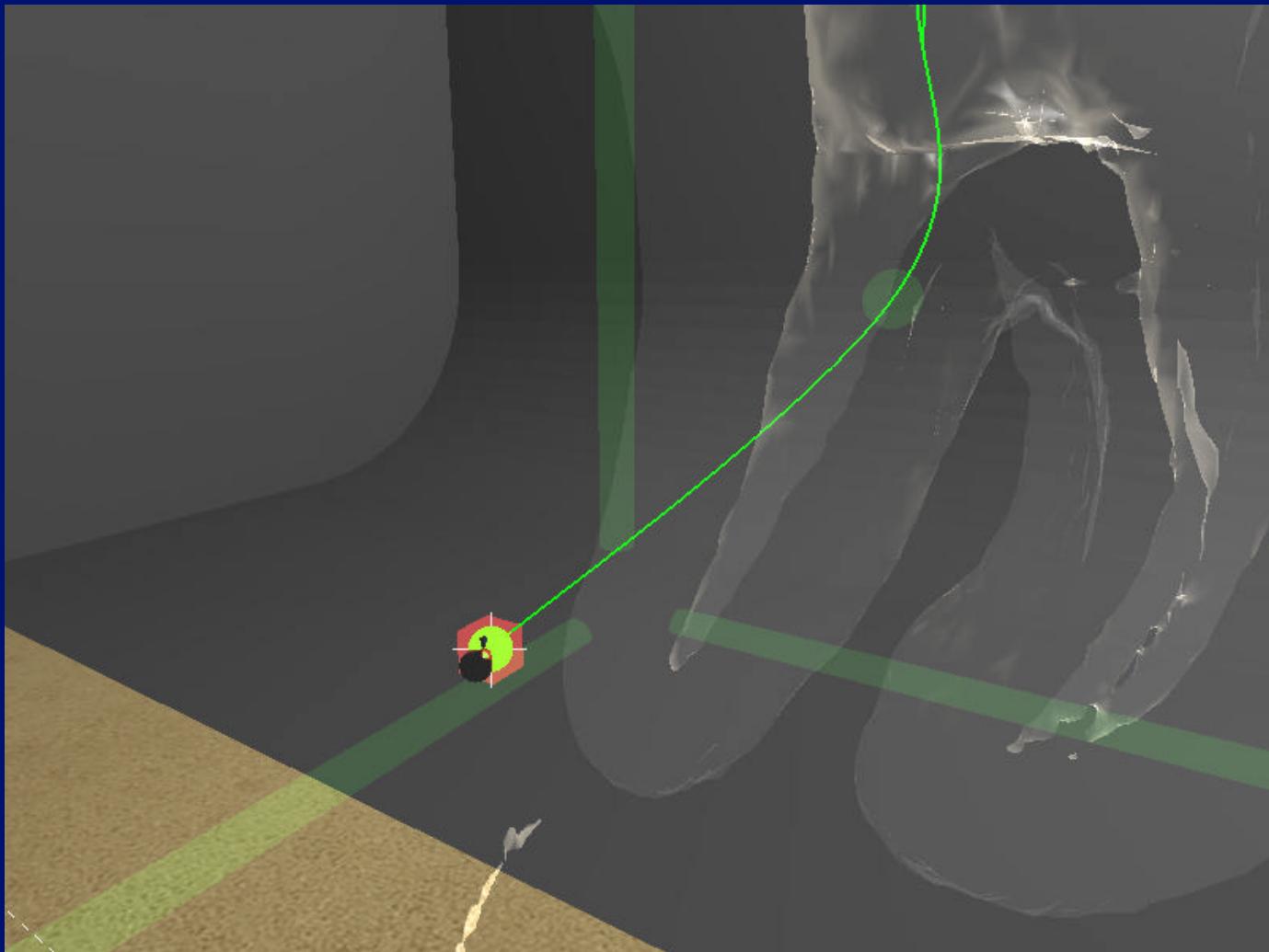
# VIRPI

- ***High-level layer on top of Aura***

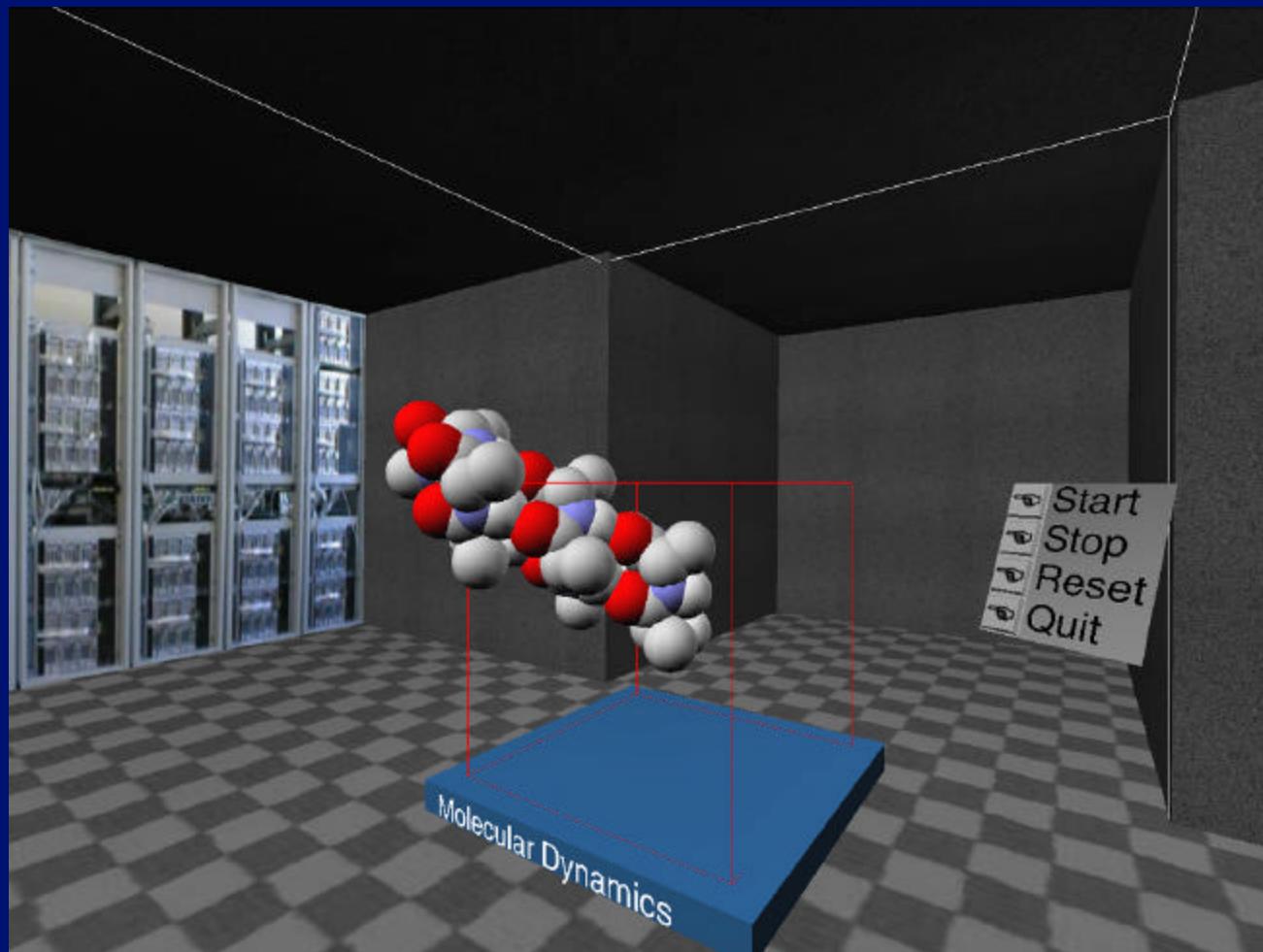
- Event tree (like 2D GUI)
- Menus, Buttons, etc.
- Manipulators (like Open Inventor)
- Way to describe measuring
- No VR expertise needed
- C++ and Python version



# Case Study: Dentistry



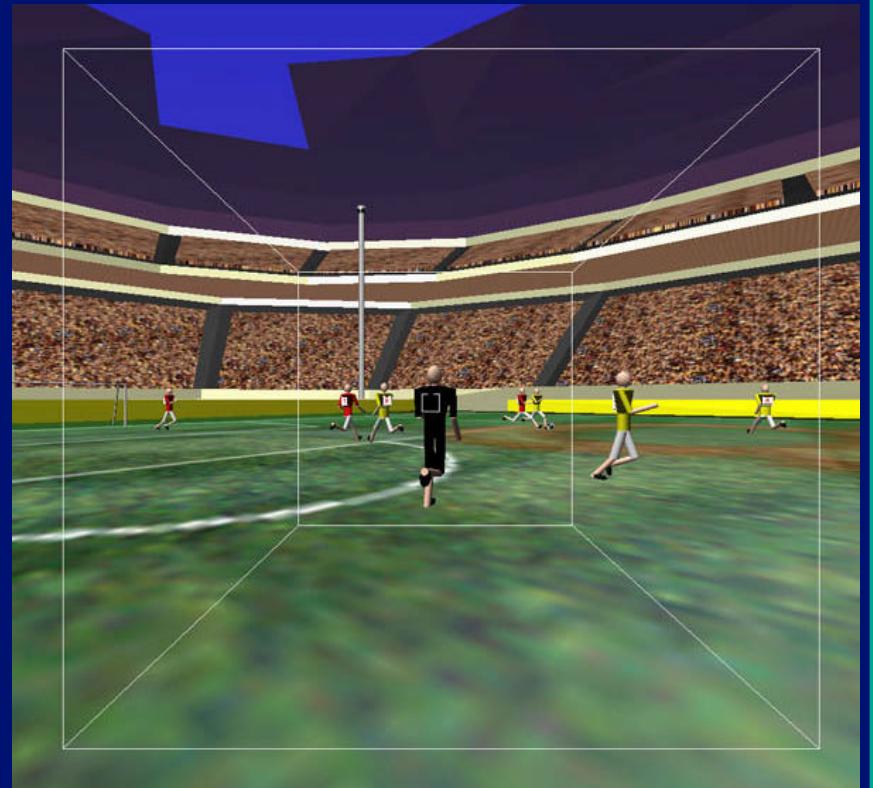
# Case Study: Molecular Dynamics





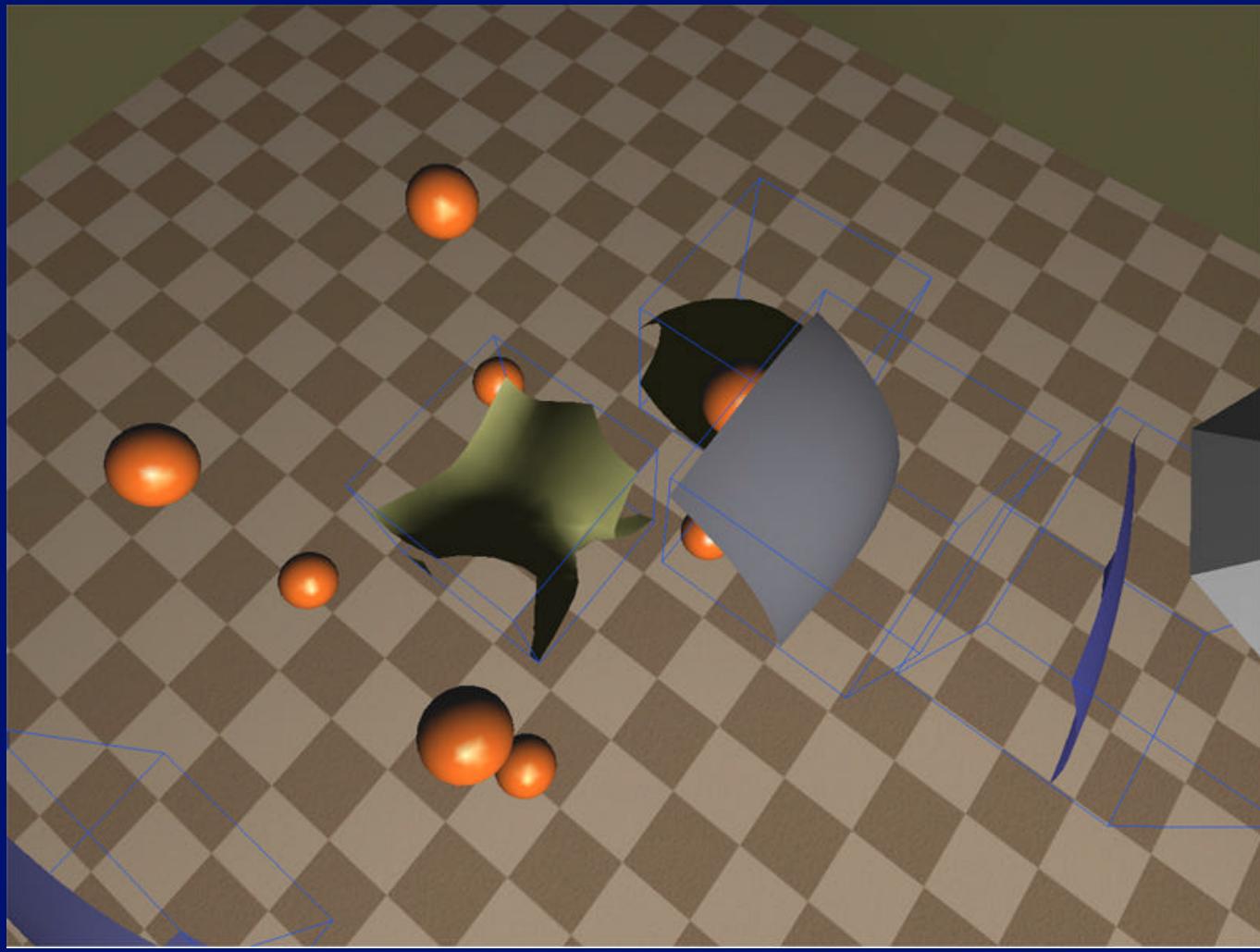
# Case Study: Robot Soccer

- *AI challenge: Multi-agent research framework*
- *Natural interaction*
- *Interactive visualization*
- *Distributed real-time collaboration*
- *Latency*



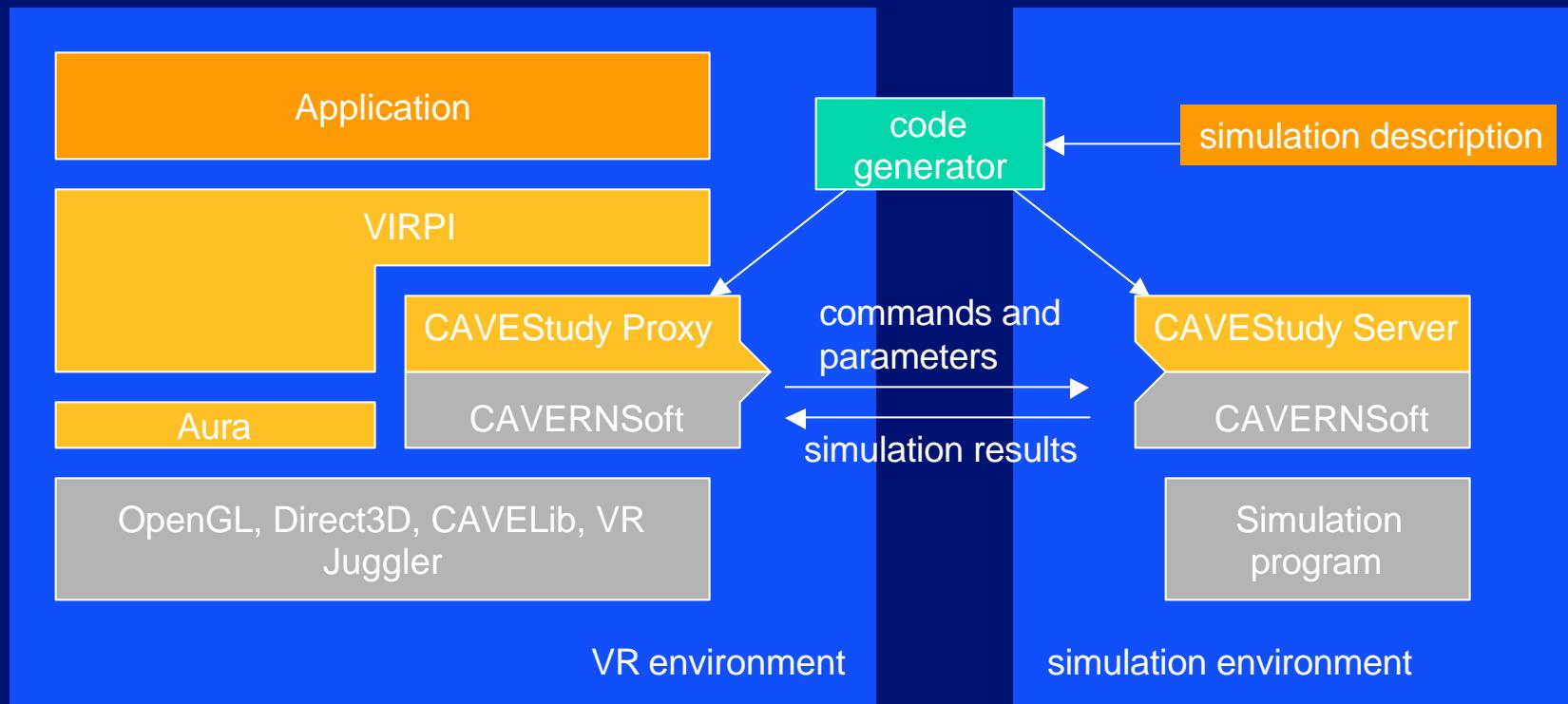


# Case Study: Electric Charges



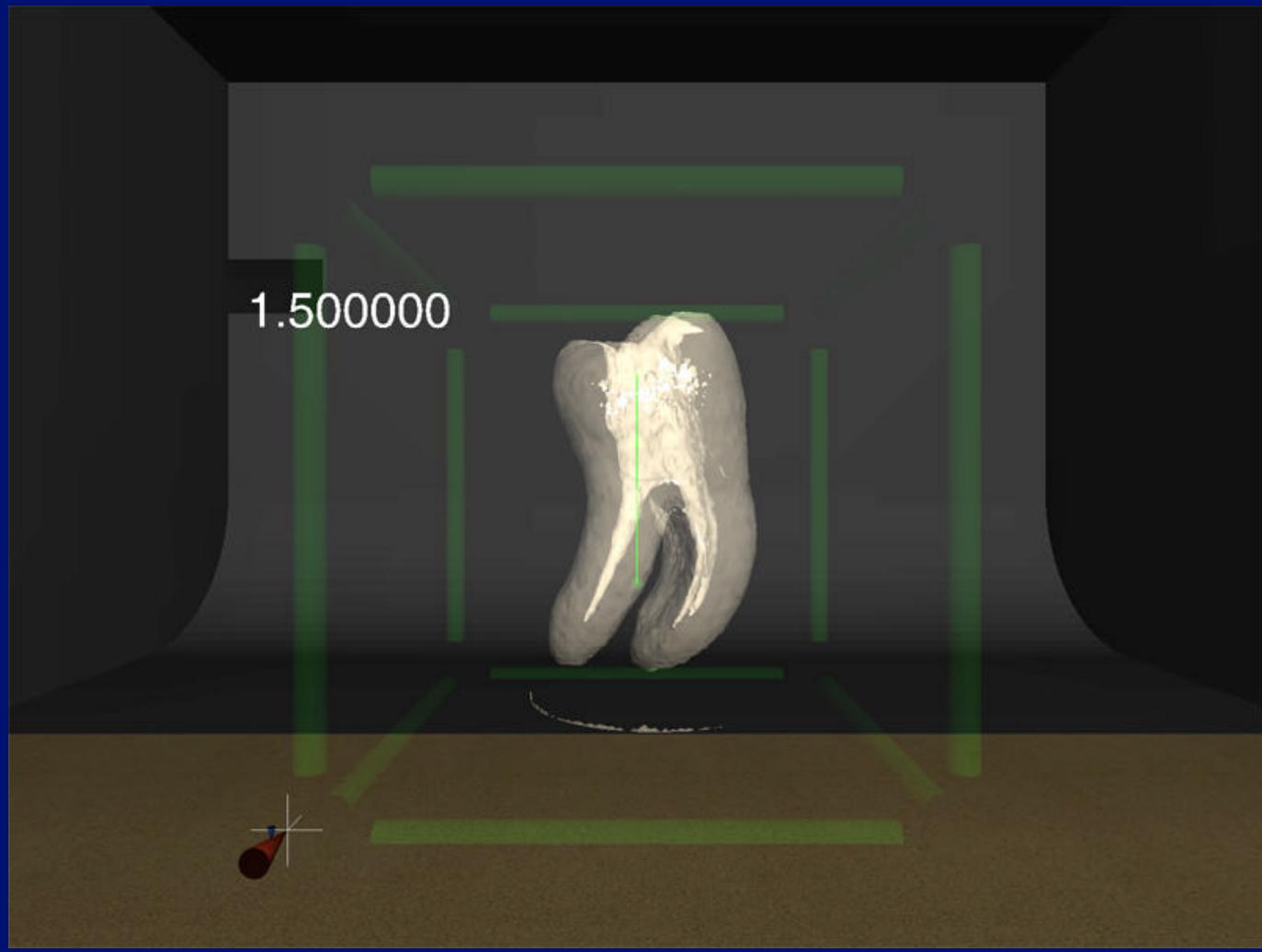


# Software Layers





# Testcase: Dental Application





# Testcase: Dental Application

**Measuring the root canal**

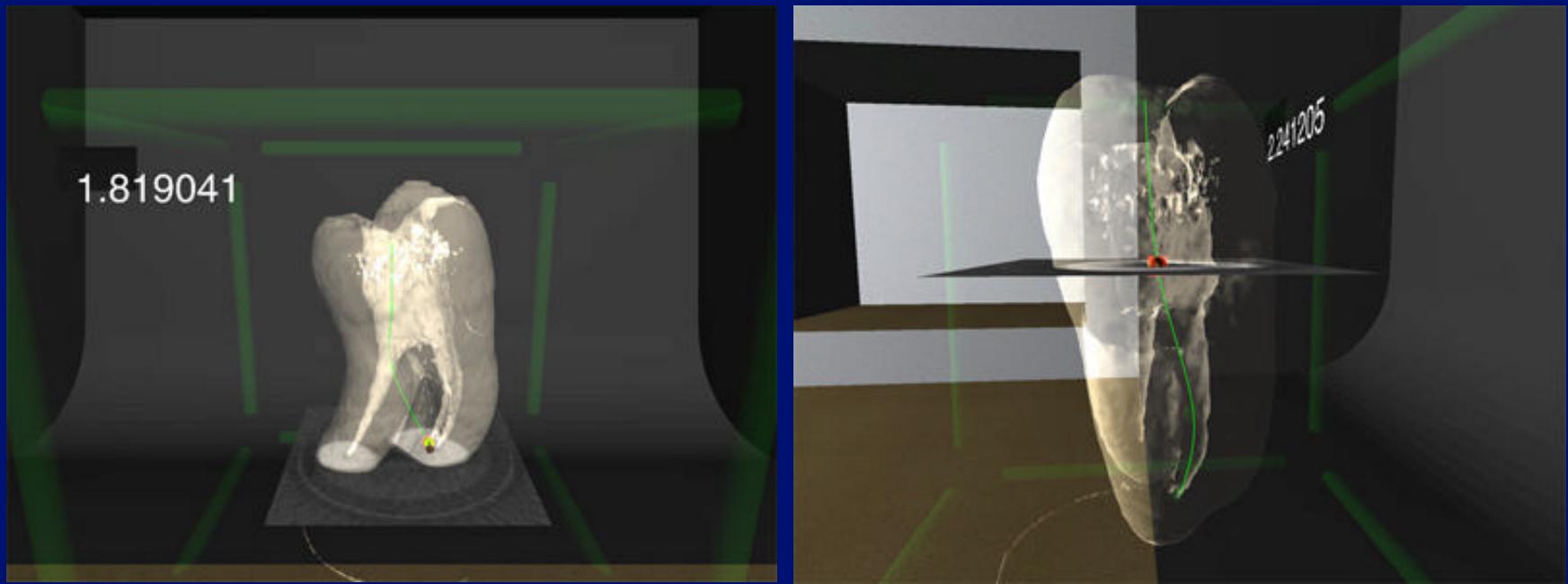
**vrije Universiteit  
FEW - ACTA**





# Testcase: Dental Application

*Measure molar root canal length with  
adjustable spline*





# Testcase: Dental Application

- ***Advantages:***

- Accuracy

- ***Disadvantages:***

- Availability
- Cost
- Calibration



# VIRPI: Measurement

- ***Select subset of data***
- ***Overload generic measuring view where:***
  - Events trigger update-method with:
    - *Access to the data*
    - *Access to selected subset description*
  - The user adds measuring code
  - Results are propagated